On the Development of Command & Control Modules for Combat Simulation Models on Battalion down to Single Item Level

Prof. Dr. Hans W. Hofmann
Dr. Marko Hofmann
University of the Federal Armed Forces Munich
Department of Computer Science
Institute for Applied Systems Science and Operations Research (IASFOR)
Werner-Heisenberg-Weg 39
D-85577 Neubiberg, Germany
e-mail: hofmann@informatik.unibw-muenchen.de

Summary: *The paper contains an **overview on the design principles and main characteristics of a family of new, strictly object-oriented combat simulation models called COSIMAC** (COmbat SImulation Model with Automated Control), developed at our Institute since 1995. They are designed as closed models which means, that a detailed modeling of the highly complicated C³I processes is indispensible. Additionally, the option of an interactive man/machine interface is implemented, which offers the possibility of manual control on different command & control levels for playing against computer generated (and controlled) forces, for experimenting with "unconventional" decisions, and for developing and improving the rule system in a trial-and-error fashion. Furthermore, the paper describes **a general architecture for the design of command & control modules**, which offers the possibility of describing tactical/operational intentions and concepts of operation in a kind of battle management language (**multilayer tactical language concept**), the **terrain representation, attrition and movement modeling**, the **development of terrain and situation assessment modules**, which are - together with a set of so-called **planning functions and spatial and procedural templates** - a prerequisite for the rule systems that generate adequate tactical missions and orders for the assigned units or simulated objects, and, finally, the **main results, conclusions** and **future developments** of the project.*

1 On the Development of the New Combat Simulation Model Family COSIMAC

1.1 Background and Main Characteristics of the New Models

As a contribution to the NATO RSG.18 study on Stable Defense (see [Hofmann et al. 95]) and in context with two doctoral thesises at our Department (see [Tolk 95] and [Schnurer 96]) more than 30,000 simulation experiments on division/brigade level have been performed with the closed, rule driven battle simulation model KOSMOS over a period of four years. They cover more than 340 different scenarios featuring, attacks by two different types of generic divisions against three different types of defending brigades under different situational conditions involving three types of terrain, two visibility conditions, up to three degrees of defense preparation and different combat modes (e.g., with or without a preceeding delaying battle). Thus, in addition to addressing the primary questions raised, e.g., by the RSG.18, the experiments offered a unique opportunity for testing a rather complex model in the light of results, leading to a continuous improvement primarily of the rule sets controlling the tactical and operational decisions.

One of the experiences we gained with the KOSMOS simulation experiments was, that - for a further substantial improvement of the rule sets that control the assigned combat and combat support units - the rule system must know the tactical/operational intentions and the concept of operation of the superior command and control level in order to react adequately. This particularily applies to cases of surprising events. Otherwise the lower units would react unadequately or - in the view of the superior command - in a rather self sufficient way.

For example, the battalion commander must know the concept of operation of the superior brigade to react adequately in the view of the brigade. In reality, the battalion commander knows that from the operational order or the context of the situation (or common held exercises etc.).

Therefore, one of the objectives of the following COSIMAC project was to describe tactical/operational intentions and options, and the corresponding concepts of operation for a set of relevant scenarios, combat modes and command levels in a computer readable way (e.g., with a kind of tactical/operational battle management language which generates spatial and procedural templates) so that the rule system could operate dynamically in accordance with the (long term) intentions of the higher command even in case of rapidly changing situational conditions and/or failure of the communication system. With this approach it should also be possible to model the

Paper presented at the RTO IST Symposium on "New Information Processing Techniques for Military Systems", held in Istanbul, Turkey, 9-11 October 2000, and published in RTO MP-049.

(so-called) German "Auftragstaktik", a special type of mission-type-tactics which offers, among others, the lower command levels a comparatively high degree of independence and flexibility in exploiting favourable opportunities.

Furthermore, the new combat simulation system should provide a higher resolution than KOSMOS, to enable modeling on the basis of physically measurable input data and thus being no longer dependent on the insertion of aggregated data (i.e., Lanchester-coefficients), that had been derived beforehand by running a high resolution simulation model. (Regarding the problems incurred in deriving Lanchester-coefficients on the basis of results obtained by running a high resolution battle simulation model see [Schaub 91].

Fig. 1.1 summarizes the main characteristics of the new combat simulation models COSIMAC in comparison with the KOSMOS model.

- Higher resolution of the battlefield (down to single weapon systems, no Lanchester-approach for attrition modeling for the major weapon systems)
- Interactive and/or closed model version at the user's disposal
- Realistic representation of military C^{3}I-processes with C^2-modules down to lower command & control levels (single item, platoon, company, battalion)
- Development of sophisticated terrain analysis modules
- Data bank oriented
- Running on PC's and workstations (hardware independent)
- Strictly object-oriented with SMALLTALK or C++

Figure 1.1: Main Characteristics of the New Combat Simulation Models COSIMAC in Comparison with the KOSMOS Model

1.2 Design of the Central Simulator

Experiences obtained in developing combat simulation systems in the past at our institute have shown that a flexible architecture requires a **strict and rigorous separation of the pure combat processes** of the basic combat (or combat support) elements (objects at the lowest level of resolution; in our models, e.g., platoons in COSIMAC-P or single weapon systems in COSIMAC-WS) **from the command and control processes which control these objects.**

A major problem was constituted in "hidden" assumptions regarding the tactical behavior of these combat objects. Actually such assumptions are often firmly connected with these objects.

In order to create a rigorous separation between different processes the architecture of the new models follows a concept of layers (see Fig. 1.2). The main layers are established by the *central simulator* and a set of *command & control modules*. Besides there exist explicit *interface layers* that serve as an additional abstraction and explicit *user interface layers*.

In the **Central Simulator** (see Fig. 1.3) the terrain, environmental data (e.g., weather data) and the basic combat (or combat support) elements as well as their associated models are being administrated. Furthermore the central simulator controls the simulation. The combat elements are described by a set of mainly physical and/or technical (input) data. The elementary processes of these objects are implemented in the associated models. These encompass reconnaissance, attrition, movement, communication, manipulation of the environment, and transport of combat elements. Furthermore, the associated models imply basic knowledge for command & control processes, e.g., target acquisition and selection, route planning etc.

The **layer of the command & control modules** consist of the C^{3}I-modules for the different C^2-authorities and/or an interactive user interface. Furthermore, the C^2-modules administrate the individual perceived situation of a C^2-authority.

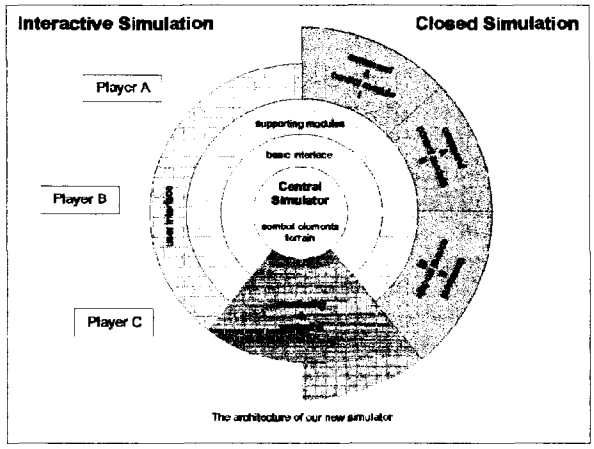

Figure 1.2: General Architecture of COSIMAC

The Central Simulator allows the access to the basic combat (or combat support) elements only by a set of elementary orders by which these elements are being controlled. This set of elementary orders is not identical with the battle management language as described in *Chapter 3*, and which will be used only between and within the command & control modules and moreover will also be much more extensive. For this very reason the command & control modules do not communicate directly with their associated combat elements but rather via a further interface object. Conversely, the communication of the combat elements addressed to their command & control modules also works via an interface. Thereby a strong logical separation between the combat elements and command & control modules can be achieved thus providing an easier exchangeability, extension and optimizing of the command & control functions. (In older models the firm

connection of command & control knowledge with the basic combat elements had revealed itself as an impasse.) Moreover, this design principle also offers the possibility of a "physical" separation between the command & control modules and the combat elements, i.e., the possibility of running the simulation on several computers is considerably facilitated.

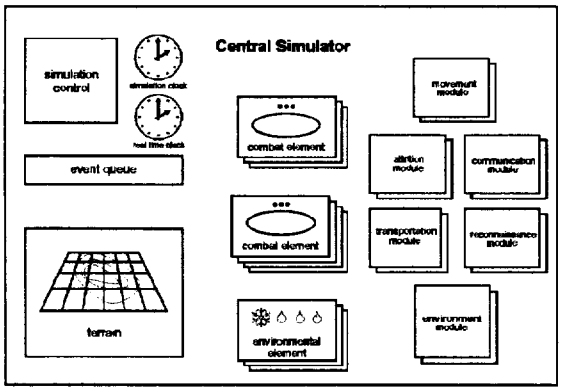

Figure 1.3: Architecture of the Central Simulator at Platoon Level (COSIMAC-P)

Nevertheless, elementary command & control knowledge has been implemented in the models associated with the basic combat elements.

The reason for this is that a strict separation of all elementary command & control processes from the basic combat elements would cause an extremely high flow of information via the interface between the Central Simulator and the command & control modules. Since, however, tactically adequate behavior of the basic combat elements is highly dependent on the specific situation and location it must be exchangeable during the running period. Therefore, a single combat element is not only allotted to one associated model for an aspect such as, e.g., target aquisition or movement, but rather to several models, one of which is the currently used one.

Since the tactical behavior of the basic combat elements used in the Central Simulator is defined by terrain to a high degree, the representation of terrain is explained in the following chapter.

1.3 Terrain Representation

For model development, implementation and testing we presently employ the digital maps used already in the BASIS simulation experiments[1]. They comprise three different pieces of real terrain in Germany and resemble the following terrain types:

- mountainous/wooded (Furth im Wald, 8*14 km^2),
- rolling hills/partly covered (Bubach, 6*10 km^2),
- flat/open (Grettstadt, 6*10 km^2).

They contain elevation, terrain vegetation and trafficability data for square grid sizes of 25, 50 or 100 m. The altitude resolution is 10 cm, resp. 1 m.

The altitude of the different combat vehicles as well as the altitude of the different terrain cells are considered when checking the line of sight. Additional visibility barriers are taken into account. These obstacles may be static (e.g., buildings, vegetation) or change dynamically during the simulation (e.g., to represent smoke screens dispensed by artillery or combat vehicles to protect themselves).

Moreover, vector data referring to roads and rivers were used in order to determine a set of ***cell transfer velocities*** that are subject to the four orthogonal directions and that rise values for each terrain cell calculated in connection with vegetation and altitude data. This enables us to generate and store these values in advance, according to mounted or dismounted basic combat elements and, if mounted, to the different types of vehicles (i.e., wheel, chain, air borne, etc.).

In addition to these static values taken mainly from the natural conditions of the regarded terrain, the *concept of cell transfer velocities* can also be extended by dynamic aspects such as obstruction and fire. This allows us to build simple but extremely flexible movement models for our basic combat elements in combat situations that are capable of considering natural terrain obstacles as well as obstructions and other artificial obstacles and moreover even obstacles caused by enemy fire.

Disregarding the cellular terrain representation on which most of the internal simulation processes are based with respect to the basic combat elements in an off-road combat mode, our display concept also permits the superimposal of pure bitmap- or vectordata-displays.

In addition to the old BASIS terrain data we have digitized the terrain of the CMTC HOHENFELS in a similar way to get the possibility of comparing some scenarios (initial situation and combat dynamics) of real held exercises in the CMTC with replayed scenarios performed with the COSIMAC models.

1.4 Attrition and Movement of the Combat Elements

Movement Modeling

[1] BASIS is a high resolution, stochastic Monte Carlo -type combat simulation model developed 1982 - 84 in PL1 for a mainframe computer at our University. It permits the detailed simulation of battalion-size ground forces defending against a sequence of regimental-size attacker forces. Resolution goes down to every combat vehicle or weapon system. It is a closed, script driven model (without C^2-models) and was extensively used for a study on „Non-offensive Defense Options" in 1984 - 85 and the derivation of Lanchester-coefficients for simulation experiments with the KOSMOS model. In 1992 it was re-implemented in PASCAL on an Apollo workstation and in 1996 in C++ on a PC. Presently it is used for test purposes, e.g., for comparing the results of the pure combat processes of the new model COSIMAC with those obtained with the BASIS model. For further information on the BASIS model see [Hofmann et al. 84].

The modeling of movement of the low level combat elements is a decisive process in every high resolution combat simulation system, especially if the system is able to conduct **automated route planning**.

Since COSIMAC is based on a square grid terrain model we first implemented the algorithms which are most commonly used to optimize routes in grid models, the Dynamic Optimization algorithm or a specific version of Dijkstra's algorithm for route planning (see, e.g., [Fould 1992]). However, in order to improve the performance of the Central Simulator we currently use a special algorithm, which turns out as a mixture of Dynamic Optimization, Dijkstra's algorithm and Branching & Bounding. The basic idea of this approach is to reduce the number of nodes (or terrain cells) permanently labelled by the spirit of Branching & Bounding by taking into account not only the time (or cost of movement) from the starting to the regarded terrain cell but also some estimate of the further distance from the regarded node to the target terrain cell. Furtheron, the possible routes of a regarded combat element (or unit) are confined beforehand by assigning mobility corridors. In other words: The combat elements can only move within predefined corridors which correspond to predefined combat sectors. Additionally, we made some research on simplified heuristic versions of this algorithm which will not find precisely the real optimum in any case, but are much faster.

In many combat simulation systems usually only one criteria is chosen for the route planning:

- time, e.g., the goal of optimization is to minimize the time a combat element needs to reach a (given) target cell.

We consider three additional criterias:

- path length, e.g., minimizing the length of the path between the current position and the target cell, taking regard of given constraints,
- concealment, e.g., maximizing the protection against sight (visual detection) given by vegetation or buildings, and
- altitude, e.g., trying to find a path that minimizes the altitude of all cells you tread on the way to the target cell. (This criteria can be understood as an attempt to maximize cover.)

These extensions and the combination of them not only allow us to automate route planning, they do also provide us with the ability to model appropriate tactical behavior, for instance to move a combat element or unit "like water flows" or to choose routes which avoid weapon engagement zones, detected or supposed mine fields, etc. Furthermore, this approach makes it possible to design a route planning algorithm for helicopters, e.g., using a modified version of the "altitude-oriented" algorithm.

Attrition Modeling

In accordance with the different classes of weapon systems the combat simulation system COSIMAC contains different kinds of attrition modeling approaches. The most important are roughly described in the following chapters:

- **Attrition model for the direct fire weapon systems**

  Regarding the direct fire weapons the attrition model in the single item version (COSIMAC-WS) operates on the (individual) single shot approach. For the platoon level (COSIMAC-P) neither a pure single-shot model on the single weapon system level with individual fire control nor an aggregated Lanchester-equation model is implemented. Since in the regarded version platoons are the basic combat element, it is assumed that all major weapon systems of a platoon fire at the same moment under the control of the platoon leader (or leading weapon system). This simplification can be justified by the fact that the fire unit of the combat troops is normally the platoon that contains similarly equipped vehicles. Keeping in mind this abstraction, the attrition model for the direct fire weapon systems can be roughly described as a multi-shot model on the combat element level. Moreover, COSIMAC is able to model unguided rockets, antitank guided missiles and fire-and-forget missiles of every range, taking regard of possible countermeasures during flight time.

- **Attrition model for the high-angle-fire**

  The model for the high-angle-fire (mortars, artillery systems and rocket launchers) operates on a single shot approach. For all kinds of targets (combat or combat support units) a special spatial distribution (spatial template) for the individual weapon systems is assumed, depending on their activity (see *Chapter 2.2*). With this approach we can base the calculation of losses caused by high-angle-fire on the commonly approved concept of lethal areas around the impact points of the fired shells.

2 C³I-Modeling

Since 1997 we have been working with emphasis on the development of first, simple command & control modules and rule sets for the single weapon system, platoon, company and battalion level. A prototype version for the combat modes *attack* and *defense* is ready for use. The next chapters describe the main design principles and first solutions.

2.1 On the Development of Terrain and Situation Assessment and Planning Modules

2.1.1 Terrain Assessment and Force Deployment Modules

Terrain assessment and subsequent force deployment are

major and indispensable tasks of every military commander. Therefore any C^2-module must be able to perform at least some of their subtasks. This is especially true and a demanding task for a detailed terrain model.

We started with the development of a simple, interactive terrain assessment and force deployment tool for defense operations on single item, platoon, company and battalion level for pieces of terrain as described before.

Input data were:

- forward edge of the battlefield (FEBA),
- boundaries of the defense area (defense sector),
- number and type of own troops/friendly forces,
- point of main defense effort,
- areas of field fortifications and target areas for supporting prearranged artillery fire.

These data are given by the op-order for defense.

In a first step, the algorithm figures out all points or areas of interest for a defender such as, e.g.:

- large, interrelated areas of open terrain, of forests and of urban (built up) areas (towns or villages),
- the rims of such areas with respect to the main defense direction, which offer defilade, protection against artillery fire and free firing zones for the low angle fire of the own combat elements (especially important for dismounted infantry),
- hills and other points or areas with good visibility conditions such as observation points or places for weapons with long range direct fire, e.g., long range anti tank positions.

In the next step the program deploys the own basic combat elements (platoons or weapon systems), which are available for the defender in the regarded defense area, into their initial defense positions on or near the FEBA (or into the security line or into positions in the depth) by considering (besides the points/areas of interest of the regarded terrain) the following items:

- type and maximum, minimum, and effective firing range of the combat elements and their effective (terrain dependent) firing zones in the proposed positions,
- the degree of overlapping between the different firing zones in order to minimize the number and size of dead firing zones,
- the point of main defense effort (where the degree of overlapping fire should be extremely high), and
- the (initial) spatial template for defense operations, which divides the defense area in an area on or near the security line, an area on or near the FEBA, an area of positions in the depth, and a rear defense zone.

Meanwhile, this module was extended and integrated into the simulator. With this module it is presently possible to assess terrain on battalion level for defense purposes and to deploy given forces to their initial defense positions. Furthermore, we extended the module to comprise also terrain assessment and force deployment for attack operations. Input data for this module are:

- boundaries of the attack area,
- line of departure,
- objective of the attack, number and type of own troops/friendly forces and reconnoited enemy troops,
- areas of friendly and (supposed or reconnoited) hostile field fortifications, and - as an option -
- one or more intermediate objective(s).

2.1.2 Situation Assessment and Planning Modules

These modules comprise a large number of mathematical functions that may be used in situation (and/or threat) assessment and operations planning.

Examples for situation assessment functions are, e.g.,

- force ratios,
- force concentrations,
- deep penetrations into the defense sector,
- open flanks, etc.

(Many of them are derived from the combat geometry.)

Examples for ops-planning functions are, e.g., estimations with respect to

- speed, space, and time requirements,
- availability of forces,
- losses,
- loss-exchange ratios for planned operations, etc.

A lot of these functions can be taken from the KOSMOS model. But there is still a considerable amount of them which must be newly developed, especially for the lower command levels. An example is elaborated in *Chapter 2.4*.

2.2 On the Development of Spatial and Procedural Templates for Generic Weapon Systems, Units and Force Stuctures, and a General Approach for the Assessment of Tactical Options

2.2.1 Definition of Generic Weapon Systems, Units, and Force Stuctures

With regard to the scope of this project to reflect on general solutions and in accordance with the object-oriented design principle of the simulation system COSIMAC we

are primarily interested in a general scenario design principle which covers a large variety of different kinds of weapon systems, unit types, and force structures. Therefore, we have applied for this project the modular force design principle of the KOSMOS simulation experiments and extended it to the lower command levels. (For more information see [Hofmann et al. 95] or [Hofmann, Hofmann 98].)

2.2.2 Spatial Templates

Spatial templates are defined as models of how objects (e.g., weapon systems, platoons, companies etc.) are positioned and oriented relatively to other objects. On all command levels, they describe (approximately) the spatial arrangement (grouping, formation) of units and sub-units depending on the state (e.g., type of combat) and situational conditions.

Examples:

- column or double column formation,
- line formation (permits excellent fire to the front),
- wedge, inverted wedge or Vee-formation ("Keil", "Breitkeil", formations used when the enemy situation is vague and the leader requires firepower to the front and the flanks),
- two-up (a formation with two elements disposed abreast, the remaining elements in rear).

Fig. 2.2 shows - as an example - the wedge formation ("Keil"). It was designed as a "pulling" template which means, that the leading 1. platoon - which advances to an objective area on the best route as described in *Chapter 1.4* - pulls the following two or three other platoons. They follow dynamically in predefined areas looking for best positions (with respect to cover or field of fire) for their own, whereas the overall heading, depth and width of the template is ordered by the company (or leading platoon).

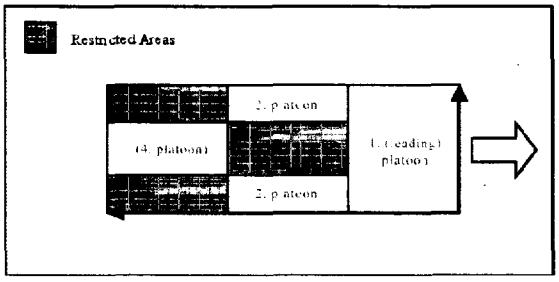

Figure 2.2: Wedge Formation ("Keil")

Fig. 2.3 depicts the inverted wedge or Vee-formation ("Breitkeil") which turns out to be a "pushing" template. In this case the 1. platoon "pushes" the other platoons before him leading and controlling them as described before on the route to the objective.

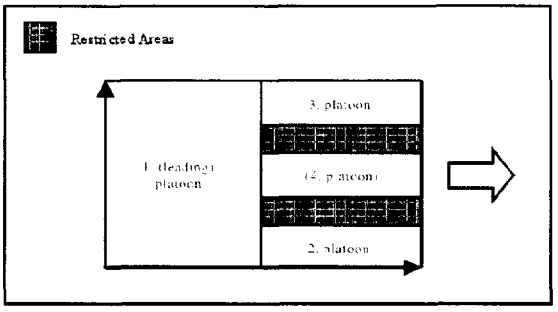

Figure 2.3: Inverted Wedge Formation ("Breitkeil")

Fig. 2.4 finally shows an example for a change of formation from the wedge over the double column to the inverted wedge formation including a change in direction of the whole formation. Intention is to avoid that the platoons interfere with others by overcrossing and/or outpacing the movements of other platoons.

2.3 Procedural Templates

Procedural Templates are defined as models on tactics, techniques and/or procedures which describe how objects or units on all command levels typically operate and work together in different combat modes (Order of Battle, employment of forces, activities, time schedules, combat dynamics, etc.).

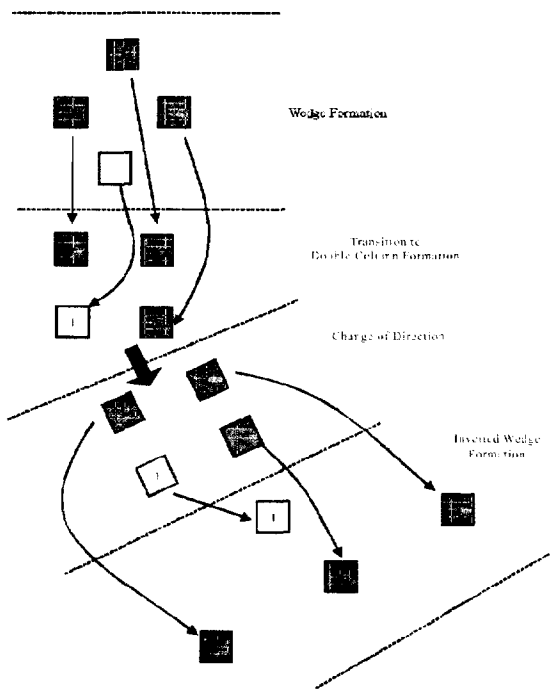

Figure 2.4: Change of Formation and/or Direction

Examples:

- leapfrogging (e.g., one combat module moves, the

other one fires (überschlagendes Vorgehen)) or

- advance in an accordion-like manner (raupenförmiges Vorgehen), but also
- schematic representations of the Order of Battle, employment of forces etc. for the different kinds and phases of an operation as described in the Field Manuals (see, e.g., FM 100-5 or HDv 231/100).

Leapfrogging and advance in an accordion-like manner are implemented on company level.

As an example for defense operations Fig. 2.5 depicts the schematic organization of a prepared defense by a *Mixed Mechanized Infantry Battalion* with two Mechanized Infantry Companies side by side in front-line positions and the Tank Company as a reserve. The scenario was elaborated by the Tactical Center of the German Army (Taktikzentrum des Herres) and published recently in [TRUPPENPRAXIS 9/97].

It shows, as an example, the "implementation" (realisation) of a given schematic organization of a prepared defence according to the respective German Field Manual (HDv 231/100) into an assumed "real" situation taking into account the perceived enemy situation, situation of own troops, terrain and a variety of further situational conditions.

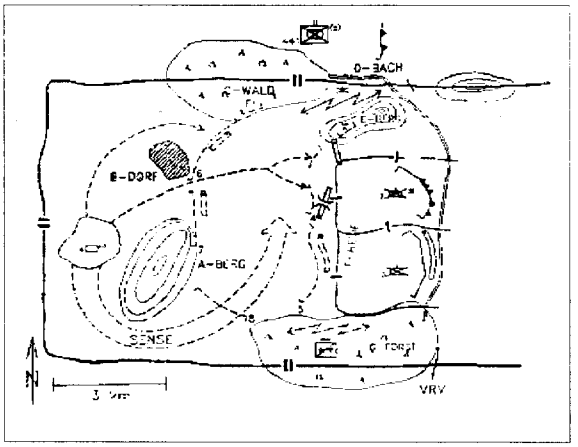

Figure 2.5: Excerpt from the Operation Plan of a Prepared Defence for a Mixed Mechanized Infantry Battalion [TRUPPENPRAXIS 9/97].

2.4 A General Approach for the Allocation of Fire and Forces and the Assessment of Tactical Options

Even though each C^2-authority and the different branches or functional staff areas have their own C^2-modules (see *Chapter 3)*, a general method for the design of C^2-modules should be mentioned at this moment: **the appliance of the multi-dimensional utility theory** for solving a large variety of decision problems from the target allocation problem up to the assessment of the effectiveness of tactical options, orders and missions, a concept that has already proven its usefulness in the KOSMOS model.

Background for this approach was the experience we made when asking military experts for rule sets for problems like, e.g., target allocation to artillery batteries, employment of reserves etc. It was very hard for them to formulate general decision rules in a precise "if - then - else" manner. Most often the answer was "that depends on the situation", specifying a set of more than 10 or 20 different influence factors.

The multi-dimensional utility theory approach reduces the general problem of fire or force allocation on all command levels to a m*n allocation problem based on a m*n-utility-matrix with each value representing the expected utility, calculated by using specific, multi-dimensional utility functions. Subsequently, the allocation of fire or forces could, for example, be made in the sequence of the utility values, taking also into account, e.g., the marginal utility.

Multi-dimensional utility criteria for the target allocation problem of artillery batteries regarding one allocation period may be, e.g.,

- expected, weighed damage (expected number of destroyed weapon systems, weighed with their values in the specific situation),
- expected effects of target suppression (often important with respect to infantry),
- cost (negative utility) of the target engagement (e.g., cost of ammunition, "cost" of being detected, etc.),
- tactical/operational aspects or urgency for allocating the target[2].

For more information see [Schnurer 96].

The corresponding criteria for the assessment of tactical options, allocation of forces, employment of reserves may be, e.g.,

- degree (or probability) of performing the given orders, missions (or intents of the higher command & control authority),
- expected weighed losses of adversary forces,
- expected weighed losses of own forces.

With this approach a large variety of different situational conditions can be considered.

However, the main problem of this approach is to get proper estimates of these (situation depending) expected values. This holds true especially for the evaluation of (given) tactical/operational options on the higher

[2] In reality, we are confronted with a very complex n-stage, 2-person-zero-sum (game theoretic) allocation problem which is far too complex for an algorithmic solution. In order to consider at least some aspects of the dynamic (or n-stage) dimension of the problem this aspect was introduced as an additional criterion. Furthermore, it offers the possibility of considering the tactical/operational intentions of the higher command in the allocation process.

command levels, at which a kind of "simulation in advance" to get these values would be indispensable.

One possibility for solving the problem would be the use of the same detailed stochastic model for the evaluation process one takes for the simulated ground truth. But that would be very (running) time consuming, especially if one considers that a large number of replications would be necessary to get expected (or mean) values. A second possibility is the development of own aggregated, expected value models for the different evaluation processes. (But this would cost a lot of time for development.)

We voted for the second option and designed and implemented

- a deterministic (expected value) model for the target allocation problem for direct and high angle fire weapons. (It operates mainly on the same input data that are also used for the simulated ground truth by the detailed Monte Carlo simulation model.)
- a comparatively simple aggregated deterministic Lanchester model, which offers the possibility for a dynamic "simulation in advance" for a limited time frame to get estimated results for the (given) tactical options to be assessed. The model operates on the perceived situation of enemy forces. (For calibration purposes it also offers the possibility of working with the real ground truth.)

Up to now, one important common principle for the design of terrain evaluation, situation assessment and planning modules within the C^2-modeling approach has been elaborated: **the appliance of classical algorithmic approaches, optimization techniques and geometric analysis as far as possible in order to exploit the advantages of modern computers for numerical solutions, to attain robust and highly efficient solutions, and to reduce the number of "if - than - else" decision rules.** But this is not sufficient. In the next chapter some further important aspects and design principles for modern command decision modeling are described.

3 On the Development of a Tactical/Operational Battle Management Language and a General Architecture for the Design of Command & Control Modules

3.1 Overview

The Tactical/Operational Battle Management Language should offer the possibility of describing (in a formalized, computer readable manner) tactical/operational intentions and concepts of operation (as described in an Operation Plan) and deliver the prerequisites for breaking down a concept of operation of a higher command level to individual missions and battle orders for the subordinate combat and combat support units, and further down to the simulated objects at the end of the command and control chain. And this should be done - as much as possible - automatically.

Taking into consideration the different levels of aggregation between a battalion operation plan and a platoon order (which are the elementary orders in our Central Simulator at platoon level), we are convinced that the main effort to realize an operation plan is connected with the task of breaking down this plan into individual missions and orders for the simulated objects.

3.2 On a General Architecture for the Design of C^2-Modules

3.2.1 Introductory Considerations

In most of the combat simulation systems realized hitherto, the modeling of the command & control process is confined to the battalion and higher command levels. This is due to the fact that on these levels analytical and geometrical approaches are (regarded as) sufficient to model such processes. But for future systems it is impossible to circumvent the modeling of at least some aspects of command & control at the company, platoon and weapon system level.

In principle, it would be possible to design a Combat Simulation System (CSS), that only copes with command & control processes at the lower echelons, but we consider that to be inappropriate, since some of the most compelling questions concerning the automation of decision making at the lower command levels are closely linked with the corresponding events at the higher levels, for instance:

- breaking down the operation plan of a battalion into concrete orders at the platoon or weapon system level,
- taking into account the higher commander's intent in unexpected situations (with disturbed communication) and
- taking advantage of favorable developments of the situation without neglecting the overall plan of the superior command level.

Therefore, we advocate for a comprehensive modeling of both lower and higher command levels (at least battalion) C^2-processes within one CSS. The nexus of all these demands is flexibility: it should be possible to replace human decision makers (man in the loop) with C^2-modules wherever you like in the simulation. This attribute is essential especially when CSS are applied within combat training exercises.

Facing these challenges we have developed a new architecture for combat simulation systems for the last four

years, that enables us to design command & control modules for each command level. This architecture is founded on the following concepts:

- separation (as far as possible) of the elementary combat processes (movement, reconnaissance, attrition, etc.) from the C^2-processes **(central simulator vs. C^2-processes concept)**,
- design of specific tactical languages for every command level and for different branches and staff functions to describe the set of options provided by the system **(multilayer tactical language concept)**,
- development of command & control modules for every command level and branch based on the corresponding tactical languages **(concept of the tailor-made C^2-modules)**,
- division of the types of combat and the general battlefield tasks into specific phases to reduce complexity **(phase concept)**,
- assignment of (a limited number of) options to each phase, which are, in a first approach, given to the system and later on generated automatically **(option concept)**,
- evaluation of these options and missions with a **generalized utility theory approach**,
- implementation of a special function to recall superior command automatons with restricted information in order to model "actions in accordance with the higher commander's intent" after failure of the communication system **(recall-function concept**, for more details see [Hofmann, Hofmann 98]),
- implementation of an **exception-interruption concept** to model a second aspect of mission type tactics: the exploitation of favorable situations, and
- strongly **object oriented programming**.

3.2.2 The Multilayer Tactical Language Concept in Detail

In *closed combat simulation systems* every tactical language (developed for a certain echelon and a certain branch) can be defined as the set of instructions, that is used to conduct the course of the battle at the corresponding level. These instructions could be simple orders, missions or even (graphical) operation plans. Thus in closed combat simulation systems the tactical language defines exactly the interface between a given echelon and its superior C^2-automaton.

In *interactive combat simulation* systems the tactical language consists of the menu of instructions, which is at the human decision maker's disposal.

Ultimate objective of the programming of a general simulation system is the exact correspondence of the "human" and the "computer" tactical language, since this is the paramount precondition for the employment of a CSS for command post exercises on different echelons without additional (service) personal. Otherwise the exchange of humans and C^2-automaton within the system in justifiable time would be impossible. Figure 3.1 shows the corresponding interface between two command levels according to the multilayer tactical language concept. (X, Y and Z designate a declining order of echelons, for instance: brigade, battalion and company.)

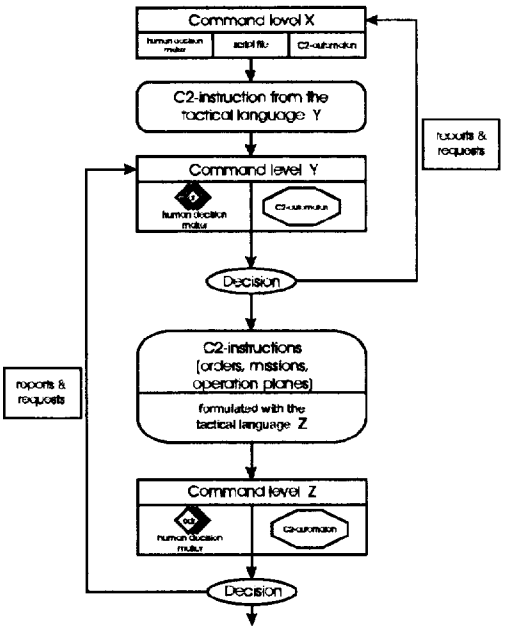

Figure 3.1: Transformation of Tactical Decisions into Concrete Instructions

This concept forces both human decision makers and C^2-automatons to formulate their *instructions* (sets of orders and missions) to subordinate units with expressions being part of the corresponding tactical language.

In general, human commanders will first make a decision and afterwards translate it into a sequence of instructions. Within an automaton the processing of the data can also lead directly to concrete instructions skipping an explicit decision.

At the lowest echelon modeled in the CSS, the decision will be transformed into a set of elementary orders, thereby controlling the elementary combat objects (platoons, weapon systems) of the central simulator.

Following this procedure, the operation plan of a unit, for instance a brigade, will be transformed into more detailed orders step by step, taking account of the capabilities and competencies of the respective echelons and finally deriving instructions to command the elementary combat objects.

The performance of such a multilayer tactical language system depends mainly on the scope of the different languages. Hence to improve the system their extension is a supreme task. Since human decision makers (military leaders) participate in the interactive version of the CSS it is also advisable to carry out this extension in a way

approaching the usual military custom.

3.2.3 The Concept of Tailor-made C^2-automatons

In general, the C^2-automatons for the closed version of the CSS cannot be designed before the corresponding tactical languages are developed. This is due to the fact that the automatons are tailored for the languages: Most of the modules of an automaton are created to perform the transition of instructions from the higher level tactical language into (more detailed) expressions of the lower level tactical language. Usually, to realize this transition a certain amount of supporting modules must be implemented. These modules include part of the tactical knowledge of a human decision maker. With no doubt, this is the most crucial step in the whole development of the simulation system, which can only be done by means of gradually improved prototypes.

To the extent the tactical languages differ from each other, the automatons will differ too. In fact, a priori there are no constraints at all for the architecture of any automaton (like, for example, a general scheme of the military decision making process); the design and implementation of the automatons depend only on the requirements of the languages.

This is why we call this approach the concept of tailor-made C^2-automatons. It provides us with the ability to design and implement a variety of different C^2-modules *within one CSS*. Therefore, such a system can be considered as a **general test environment for the development of C^2-automatons.**

3.2.4 Modeling Command & Control on the Battalion Level

In the following a view is given on the basic modules of a decision making automaton at the battalion level. Although it can only be a first sketch, we think that this list gives a helpful guideline. What we need is:

- a mission analysis module (considerably simplified by the tactical language concept, since it only searches for key words to trigger the assessment modules),
- an intelligence estimation module which builds up the perceived situation and must finally encompass a comprehensive enemy situation and threat assessment,
- a module to analyze and project the own tactical situation,
- a module to analyze and project the own logistical situation,
- a terrain assessment module,
- a module that administers pre-developed options (courses of action) or generates these options by itself according to the different phases of combat, and
- an explicit decision making module, which connects all the information provided by the evaluation modules with the different options, weighs these options up and eventually chooses one of them for a decision.

Figure 3.2 shows how these modules can constitute a prototype of a battalion C^2-automaton.

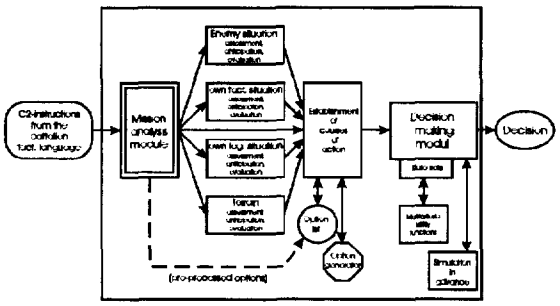

Figure 3.2: Possible Architecture for a Battalion C^2-automaton

Of course this is only one exemplary solution, but it certainly comprises the main elements necessary to model military decision making at the battalion echelon in general.

3.2.5 The Transformation of the Battalions Commander's Intent on the Company Level

The Company Tactical Language

The company tactical language (CTL) mainly serves to translate the battalion commander's decision into company level instructions. Thus the question is: Which aspects of combat are usually performed at the company level, respectively: What are the corresponding orders to transform the commander's decision?

Above all the company is responsible for the *immediate coordination* of fire and movement - even when facing enemy fire and difficult terrain - and all the arrangements to perform it. Consequently, the company tactical language must allow the battalion commander to set the stage for this coordination which means that the CTL will mostly comprise the following instructions:

- a concrete *movement order*, if needed specified with
  - an objective,
  - if need to be: intermediate objectives,
  - a line of movement,
  - a velocity,
  - an exact time to start the movement
  - an order and maybe
  - a formation,

- a *fire control order* (addressing chiefly the clearance of fires) and
- a *communication order* (inevitable for the reward passage of lines).

A next step to extend the scope of a general CTL surely includes:

- an *emplacement order*, subdivided in
  - a detailed fighting position order and
  - an order to occupy a battle area,
- a set of *logistical orders* to organize supply and maintenance,
- an *order to attach/detach platoons*,
- an *order to allot sectors of observation and fire* and
- a set of *orders to command the combat of dismounted and mounted fighting forces*.

With this set of orders it should be possible to "translate" most of the battalion commander's decision into concrete instructions for the company.

Realization of the Company Tactical Language Instructions within the Company C²-automaton

As mentioned before decision making at the company level differs markedly from its higher level counterparts. Instead of being driven by a general information processing leading to a choice among different options, low level command resembles frequently a permanent adaptation to the current situation, using proven actions and arrangements.

Therefore, a straightforward processing of the battalion orders is seldom feasible. A very instructive example for this difficulty is the tactical movement appropriate to terrain and situation. Without a notion of the combat formations (wedge, column, Vee-formation, etc.) and the possibilities to advance (by bounds, by echelon, leapfrogging or accordion-like) any C²-automaton will fail to produce reasonable commands for the platoon level.

In order to solve this problem we endowed the company automaton with a set of supporting modules, reflecting exactly this kind of skill and knowledge. Since calling these modules is quite similar to calling the whole automaton with tactical language instructions, we have named the totality of these modules the *company tactical realization language* (CTRL). It is essential to notice that the orders triggering these support modules are not part of the company tactical language (CTL), which implies that they cannot be used at the battalion level to specify the battalion commander's decision. In fact, they are not accessible for him at all (see Figure 3.3). Notwithstanding this constraint the object-oriented design of the simulation system permits to reuse most of the elements of the tactical realization language in different automatons.

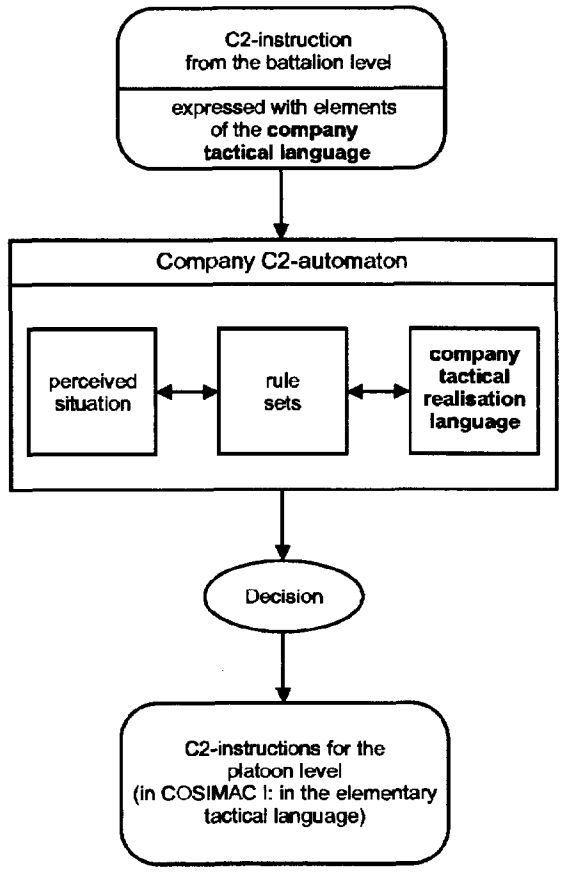

Figure 3.3: Company Tactical Language and Company Tactical Realization Language

4 Main Results, Conclusions and Future Developments

One of the main efforts is the development of robust and highly efficient rule sets for combat simulation systems (CSS) with automated control. For higher command levels we realized this task with the CSS KOSMOS. During the last five years we focused our attention to lower echelons and designed and implemented the COSIMAC models with a completely new object oriented architecture in order to evaluate different command & control automatons within one simulation system.

Among others, we have

- developed a concept for the description of tactical/operational intentions and concepts of operation with a battle management language depending on the different command & control levels and branches or functional staff areas,
- developed and implemented an architecture for the design of C²-modules to break down these concepts into individual mission and battle orders for the subordinate combat and combat support units,

- developed and implemented some detailed modules for route planning, terrain assessment for defense and attack operations, and some spartial and procedural templates on battalion, company and platoon level,
- implemented a general algorithm based on a multi-dimensional utility theory approach for solving the general allocation problem of fire and forces in order to come on robust and highly efficient solutions, and to reduce the complexity and the number of "if - then - else" decision rules.

Altogether, we come to the conclusion that the development of C^2-modules at different command levels within one combat simulation system seems to be a feasible task; even sophisticated aspects of mission-type-tactics like

- acting in accordance with the higher commander's intent or
- the exploitation of favorable unanticipated situations

are not out of reach of modern combat simulation systems with automated control.

One of the main problems of this approach is not only the complexity of the problems to be solved, which reveals the limits of what seems possible today. In our view, another major problem simply consists in the enormous amount of work, which leads to the real limits.

But the project is going on. Our MoD was pleased with the concept and decided to support the further development of COSIMAC as a research and study project, that concentrates on command decision modeling for different combat modes on the battalion, company, platoon and single weapon system level.

In the long run we think about a PC-based training or a risk evaluation and decision support tool, by means of which officers may test a range of tactical options in assumed scenarios in training or real combat situations providing them a better understanding of the regarded tactical/operational situation and assisting their decision making. More details on the project are documented in [Hofmann, Hofmann 98] and [Hofmann 00].

References

[FM 100-5] Department of the Army: Operations. Washington, DC, June 1993

[Foulds 92] Foulds, L. R.: Graph Theory Applications. Springer-Verlag, New York, 1992

[HDv 231/100] BMVg, FüH I 6: Das Panzergrenadierbataillon. Bonn, März 1988, VS-NfD

[Hofmann et al. 86] Hofmann, H.W., Huber, R.K., Steiger, K.: On Reactive Defense Options - A Comparative Systems Analysis of Alternatives for the Initial Defense Against the First Strategic Echelon of the Warsaw Pact in Central Europe. In Huber (Ed.): Modeling and Analysis of Conventional Defense in Europe (pp. 97 - 140). New York, 1986

[Hofmann et al. 92] Hofmann, H.W., Rochel,T., Schnurer, R., Tolk, A.: KOSMOS - Ein Gefechtssimulationsmodell auf Korps-/Armee-Ebene (Version 3.0). Band 1: Beschreibung des Gefechtsmodells. IASFOR-Report Nr. S-9208, Fakultät für Informatik, Universität der Bundeswehr München, Neubiberg, 1992

[Hofmann et al. 95] Hofmann, H.W., Schnurer R., Tolk, A.: Kosmos Simulation Experiments on Stable Defense. In: Christensen T. (Ed.): Stable Defense - Final Report. Appendix 3 to Annex IV to Technical Report AC/243 (Panel 7) TR/5. NATO RSG 18, Brüssel, 1995

[Hofmann, Hofmann 98] Hofmann, H.W., Hofmann, M.: Formal Description, Modeling and Simulation of Tactical/Operational Intentions and Concepts of Operation - Final Report. IASFOR-Report Nr. S-9803, Fakultät für Informatik, Universität der Bundeswehr München, Neubiberg, September 1998

[Hofmann 00] Hofmann, M.: Zur Abbildung von Führungsprozessen in hochauflösenden Gefechtssimulationssystemen. Dissertation, Fakultät für Informatik, Universität der Bundeswehr München, Neubiberg, 2000

[Rochel 90] Rochel,T.: Zur Architektur geschlossener Gefechtssimulationsmodelle höherer Abbildungsebene unter besonderer Berücksichtigung der Modellierung und Implementierung von Führungsprozessen. Dissertation, Fakultät für Informatik, Universität der Bundeswehr München, Neubiberg, 1990

[Schaub 91] Schaub, T.: Zur Aggregation heterogener Abnutzungsprozesse in Gefechtssimulationsmodellen. Dissertation, Fakultät für Informatik, Universität der Bundeswehr München, Neubiberg, 1991

[Schnurer 96] Schnurer, R.: Zur Abbildung von Führungsprozessen in geschlossenen Gefechtssimulationsmodellen. Dissertation, Fakultät für Informatik, Universität der Bundeswehr München, Neubiberg, 1996

[Tolk 95] Tolk, A.: Zur Reduktion struktureller Varianzen - Einsatz von KI in geschlossenen Gefechtssimulationsmodellen. Dissertation, Fakultät für Informatik, Universität der Bundeswehr München, Neubiberg, 1995

[TRUPPENPRAXIS 9/97] Taktikzentrum des Heeres: Damit die Zeit nicht davonläuft (Teil1) - Beurteilung der Lage und Entschluß anhand von Leitfragen. In: TRUPPENPRAXIS 9/97